OULUN AMMATTIKORKEAKOULU

Jani Kamsula

KASSAVIRTAENNUSTEEN OHJELMOINTI

KASSAVIRTAENNUSTEEN OHJELMOINTI

Jani Kamsula
Opinnäytetyö
Syksy 2014
Tietotekniikan koulutusohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietotekniikan koulutusohjelma, ohjelmistokehityksen suuntautumisvaihtoehto

---

Tekijä(t): Jani Kamsula
Opinnäytetyön nimi: Kassavirtaennusteen ohjelmointi asiakasyrityksen tarpeisiin
Työn ohjaaja(t): Eero Nousiainen
Työn valmistumislukukausi ja -vuosi: syksy 2014
Sivumäärä: 32

Opinnäytetyönä tuli toteuttaa kassavirtaennusteohjelmistokomponentti Piimega Oy:n Piimega Total -toiminnanohjaus- ja asiakkuudenhallintaohjelmistoon. Kassavirtaennusteen avulla voidaan esittää tietoa tulevista kassatapahtumista ja selkeyttää yrityksen kassan tilannetta sen käyttäjille.

Opinnäytetyössä pyrittiin hyödyntämään Piimegan yleisesti käyttämiä ohjelmointi tekniikoita ja määrityksiä, jotta voitaisiin varmistaa kassavirtaennusteen integraatio eli yhteensopivuus yrityksen muiden tuotteiden kanssa.

Opinnäytetyö aloitettiin suunnitteluosuudella, jossa käytiin läpi, miten ohjelmistokomponentti kannattaisi järkevimmin rakentaa, jotta suuremmilta sudenkuopilta voitaisiin välttyä. Suunnitteluosuuden aikana luotiin myös säännöstö siitä, kuinka tietoa tulee käsitellä, jotta mahdolliset integraatiot tapahtuisivat jouhevasti. Vaatimusten selkeydyttyä ryhdyttiin ohjelmistokomponenttia kehittämään vesiputousmallin mukaisesti. Ohjelmointityön jälkeen viimeiseksi osuudeksi jäi laajojen SQL-kyselyiden laatiminen asiakasyrityksen tarpeiden mukaisesti, käyttäen suunnitteluvaiheessa luotua säännöstöä, jonka avulla varmistetaan dynaamisen tietojenkäsittelyn toimivuus.

Asiasanat: Taloushallinto-ohjelmat, Tietokannat, Tietokantaohjelmointi, ASP.NET, Visual Basic, Telerik, Ajax, SQL

ALKULAUSE

Haluaisin kiittää Piimega Oy:tä loistavasta mahdollisuudesta hioa ohjelmisto-suunnittelutaitojani. Erityisesti haluan kiittää Toimitusjohtaja Vesa-Pekka Palokangasta ja vanhempaa ohjelmoijaa Tuomo Ranuaa heidän väsymättömästä ohjauksestaan.

Oulussa 27.11.2014

Jani Kamsula

SISÄLLYS

SANASTO

.NET framework	.NET framework on Microsoftin kehittämä ohjelmistokomponenttikirjasto, jota Microsoft Visual Studio.net -ympäristöön ohjelmoidut ohjelmat käyttävät. Se tukee n. 20 ohjelmointikieltä, joista suosituimmat ovat C# ja VB.net. (1; 2.)
Ajax	Akronyymi sanoille Asynchronous JavaSrcipt And XML. Käytännössä Ajax on ryhmä toisiinsa sidoksissa olevia webohjelmointitekniikoita, joilla pyritään antamaan käyttäjälle asynkroninen käyttökokemus, jolloin sivu lataa vain tarvittavat asiat serveriltä sivulle.(3, s. 39; 4.)
ASP.NET	ASP.NET on Microsoftin kehittämä web-ohjelmistokehys joka on osa .NET FrameWorkia. Sen avulla voidaan luoda dynaamisia web-sivuja, web-ohjelmia ja web-palveluja. (5, s. 3–5.)
Client side, asiakaspuoli	Client sidella viitataan tapahtumiin, jotka tapahtuvat asiakkaan puolella ohjelmistossa (6).
ERP-ohjelmisto	ERP-ohjelmistolla tarkoitetaan toiminnanohjausohjelmistoa. ERP on lyhenne sanoista Enterprise Resources Planning, eli yrityksen resurssien hallinnan suunnittelu. (7.)
for each -silmukka	For each -silmukka on toistorakenne, jossa toistetaan tapahtuma jokaista kokoelmassa (lista, tietokantamuutuja, tms.) olevaa tapahtumaa kohden. (8.)
ISO8601,ISO	ISO8601 on kansainvälisen standartoimisjärjestön (ISO) antama standardi päivämäärän ja ajan esittämistavasta (9).
Integer, Int	Integer on kokonaislukumuuttuja, tässä kyseisessä tapauksessa Int32. Int32 pystyy pitämään sisällään positiivisia ja negatiivisia kokonaislukuja. Int32:ssa 32

	tarkoittaa 32-bittiä ja pystyy pitämään käsittelemään numeroita aina välillä -2 147 483 648 ja 2 147 483 647. (10; 11.)
JavaScript, JS	JavaScript on pääasiassa web-ympäristössä käytettävä komentokielisarja. Sen tärkeimpiin ominaisuuksiin kuuluu kyky luoda web-sivuille dynaamista sisältöä. (12.)
Kassavirtaennuste	Kassavirtaennuste on keino seurata yrityksen kassan kehitystä annetulla aikavälillä. Se koostuu yrityksen tiedetyistä tuloista ja menoista, joita kassavirtaan laskemalla saadaan annettua realistinen kuva kassan kehityksestä.
Mark-up, mark-up language	Mark-up language tarkoittaa kuvauskieltä Kuvauskielillä määritellään muotoiluja web-sivustoille, ja yleisin näistä on HTML (13.)
Microsoft SQL Server, MS-SQL	Microsoft SQL Server on Microsoftin kehittämä relaatiotietokannan ohjausjärjestelmä. Sen tarkoitus on tallentaa tietoa ja mahdollistaa sen lataaminen toisille ohjelmistosovelluksille. (14.)
Microsoft Visual Studio	Microsoft Visual Studio on Microsoftin ohjelmistonkehitysohjelmisto (15).
Modal ikkuna	Modal ikkuna on lapsi-ikkuna, joka vaatii käyttäjän interaktiota ennen paluuta isä-ohjelmaan (16; 17.)
Nvarchar	Nvarchar on SQL-tietokannoissa esiintyvä tekstityyppinen muuttuja, johon voidaan tallentaa tekstiä (18).
Parsin	Parsin on prosessi, jossa muokataan ja lajitellaan tietoa olemassa olevan säännöstön mukaisesti toisenlaiseen muotoon (19).
Primääri-avain	Primääri-avaimella tarkoitetaan tässä yhteydessä uniikkia tunnistetietoa, jonka avulla voidaan sitoa isä- ja lapsitaulujen rivejä yhteen (20).
Postpack	Postpack on tapahtuma, jolla lähetetään tietoa osoitteeseen, joka on sama kuin sivulla, josta tiedot on lähetetty. Tämän

	jälkeen palvelin päivittää sivun käyttäen kyseisiä tietoja. (21.)
Piimega Total, Total	Piimega Total on ERP-, eli toiminnanohjausohjelmisto. Se koostuu ohjelmistokomponenteista, joista asiakas valitsee tarvitsemansa. (7.)
RadButton	RadButton on Telerikin luoma ASP.NET -komponentti, joka toimii nappina (22).
RadHtmlChart	RadHtmlChart on Telerikin luoma ASP.NET -kontrolli jossa voidaan esittää tietoa graafisessa muodossa (23).
RadGrid	Radgrid on Telerinkin luoma ASP.NET-kontrolli, jossa voidaan esittää tietoa ruudukkomuodossa. Kontrollia on yleisesti käytetty esim. tietokannasta haetun tiedon esittämiseen. (24.)
Server side, serveripuoli	Server sidellä tarkoitetaan ohjelmistossa palvelimen puolella tapahtuvia asioita (25).
SQL	SQL eli Structured Query Language on standardoitu kyselykieli, jolla voidaan tehdä hakuja relaatiotietokantoihin (26, s.1).
Telerik aspnet-ajax	Telerik aspnet-ajax on graafisten käyttöliittymien kontrollikirjasto ASP.NETille ja Ajaxille (27).
Telerik ASP.NET panel, panel	Panel on tapa sitoa objekteja yhteen niiden käsittelyä varten (28).
VB.NET	Visual Basic on korkean tason ohjelmointikieli jota käytetään Microsoft .NET Framework:issä (29; 30).
while-silmukka	While-silmukka on toistorakenne mm. VB.NET:issä. While-silmukassa toteutetaan toistoja, kunnes ohjelmistoon määritetty ehto on täyttynyt. (31.)

1 JOHDANTO

Suoritin opinnäytetyöni Piimega Oy:ssä, joka oli minulle jo ennestään tuttu paikka. Aluksi suoritin siellä yrityslähtöiset harjoitteluni ja myöhemmin siirryin myös yrityksen palkkalistoille kesätöiden kautta. Opinnäytetyön tekeminen yritykselle oli luonnollinen jatkumo hyvälle yhteistyölle. Hyvä kehitysympäristön ja asiakasympäristön tunteminen olivat luonnollisesti tärkeitä valtteja projektin toteutuksen kannalta.

Opinnäytetyöni tarkoituksena oli luoda uusi ohjelmistokomponentti, joka tukisi jo olemassa olevia komponentteja Piimega Total -ohjelmistossa. Komponentti itse on kassavirtaennuste, ja sen päätarkoituksena on havainnollistaa jo olemassa olevaa dataa helposti luettavaan ja selkeään muotoon.

Total on resurssienhallintajärjestelmä ja raha on nykymaailmassa yrityksen tärkein resurssi. Luonnollisesti rahavirtoja halutaan seurata samoin kuin muitakin resursseja, olivatpa ne sitten vaikka miestyötunteja tai asennusmutterien määrää varastossa. Tästä syystä Piimega haluaa tarjota asiakasyrityksilleen tavan seurata rahan liikettä ns. reaaliajassa, antaen asiakasyrityksen johtoportaalle tärkeitä tietoja tulevaisuuden suunnittelua varten.

Ohjelmistokomponentti tuli toteuttaa käyttäen yleisiä menetelmiä, niin että se voidaan helposti integroida yhtiön muiden ohjelmistojen kanssa. Kassavirtaennuste toimii lähes täysin SQL-tietokannan tietojen pohjalta. Koska tietokantojen sisältö luonnollisesti eroaa eri ohjelmistojen ja asiakkaiden välillä, tulee tietojen esityksen rakentua dynaamisesti staattisten hakumääritteiden ympärille. Staattisen tiedon esitys olisi näiden muuttujien takia looginen mahdottomuus. Toisin sanoen ohjelmistokomponentti rakentaa tiedon esityksen ajon aikana käyttäen hyväkseen tietokantataulujen suunnitteluvaiheessa luotuja hakurakenteita, jotka määrittelevät hakutulosten nimien muodot ennakoitavaan muotoon.

2 TEKNIIKAT

ASP.NET

ASP.NET on ASP:n seuraava sukupolvi, joka mahdollistaa tehokkaan palvelinpuolen ohjelmoinnin ja yksinkertaistaa monia aikaisemmin monimutkaisia asioita. Versiosta 3.5 lähtien ASP.NET on ollut kokonaisvaltainen kehitysalusta Ajaxia hyödyntäville web-sovelluksille. (5, s. 3–5)

Visual Basic.NET

Visual Basic.NET on seuraavan sukupolven versio Visual Basic -ohjelmointikielestä. Se kuitenkin poikkeaa huomattavasti aikaisemmista sukupolvista. Microsoft on tehnyt siihen muutoksia, jotta se toimisi paremmin .NET-ympäristössä ja olisi varteenotettava vaihtoehto muille ohjelmointikielille. Näistä muutoksista johtuen VB.Net on tehokas oliopohjainen ohjelmointikieli. (30, s. 13)

SQL

SQL on lyhenne sanoista Structured Query Language, rakenteellinen kyselykieli. Tämä on nykyisin de facto -standardin asemaan noussut relationaalisten tietokantojen hallinta- ja käsittelykieli, joka kattaa tietokannan määritykset, käsittelyn ja käytön ohjauksen määritysten avulla. (26, s1).

SQL avautuu parhaiten mielestäni esimerkin kautta:

SELECT ID, Etunimi + ' ' + Sukunimi AS Kokonimi FROM Tyontekija WHERE Toimipaikka = OULU AND tyopiste = varasto AND Senior = True

Tässä on esimerkki, jossa haetaan Työntekija-taulusta ID, työntekijän Etunimi ja Sukunimi. Etunimi ja Sukunimi yhdistetään yhdeksi kentäksi, jonka nimenä on Kokonimi. Etu- ja Sukunimen välissä on välilyönti.

Haku suoritetaan Työntekija-tauluun ja valintaehtoina on, että työntekijän tulee olla Oulusta. Työntekijän tulee olla varastotyöntekijä ja hänen tulee myös olla Senior eli vanhempi työntekijä.

Ajax

Ajax tulee sanoista Asynchronous JavaSrcipt And XML. Ajax ei itsessään ole tekniikka, vaan joukko tekniikoita, joita käytetään luovalla tavalla mahdollistamaan web-sivujen dynaaminen lataaminen osa kerrallaan. Edellä mainittu poistaa tarpeen ladata koko sivu yhdellä kertaa. Näillä tekniikoilla saavutettava lopputulos ei ole kuitenkaan mitään uutta. Asynkronisen sivulatauksen historia voidaan jäljittää niinkin kauas kuin Internet Explorer 3:een asti. (3, s. 39; 4.)

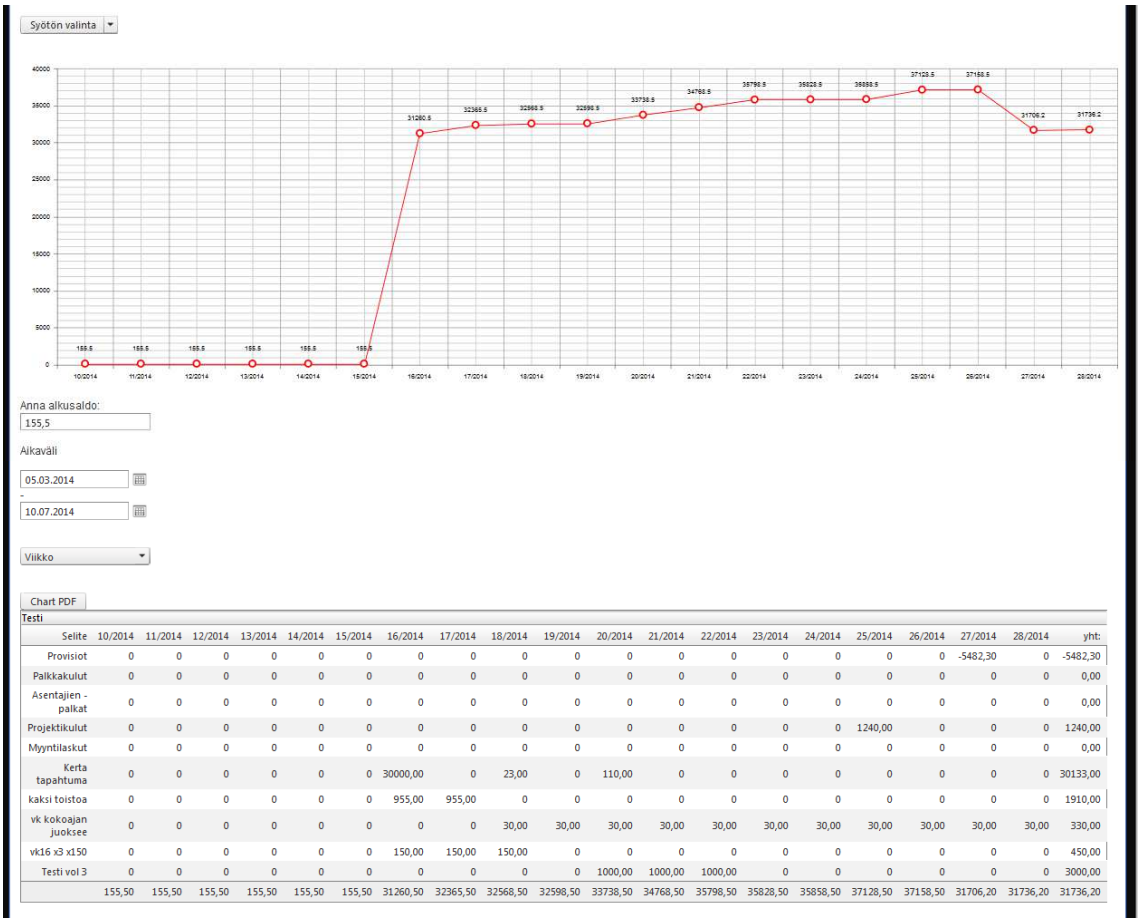
3 KASSAVIRTAENNUSTE

Kassavirtaennusteen rakentaminen aloitettiin pitämällä ensimmäinen suunnittelupalaveri. Palaverissa käytiin läpi ohjelmistolle asetettavia vaatimuksia sekä mahdollisia teknisiä ongelmakohtia (mm. tietokantarakenteet). Näiden annettiin hautua hetki, ennen kuin pidettiin seuraava suunnittelupalaveri, jossa lyötiin lukkoon komponentin vaatimuksia ja tekniikoita.

3.1 Ulkoasu

Määrittelin, että kassavirtaennusteen pääsivulla tulisi olla RadHtmlChart (23), jossa esitetään piirroksena kassavirtaennusteen tapahtumia. Sen alle tulisi sijoittaa RadGrid (24), jossa esitetään vastaavat tiedot yksityiskohtaisemmin. Molempien komponenttien tiedon esitys tapahtuisi päivä-, viikko- ja kuukausitasolla. (Kuva 1.) Myöhemmin päivätasosta kuitenkin luovuttiin sen suhteellisen tarpeettomuuden vuoksi. Esimerkiksi kahden kuukauden otteessa saattoi olla kuusikymmentä tyhjää päivää, ja vain yksi merkinnällinen päivä.

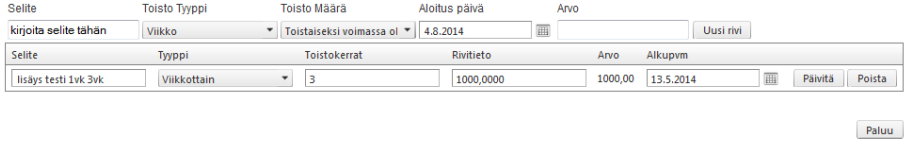
Tiedonsyöttö päätettiin toteuttaa käyttäen modaaleja pop-up-ikkunoita, joiden kautta voidaan syöttää ja muokata tapahtumia (kuva 2, kuva 3, kuva 4). Tarkempi tiedonesitys päätettiin myös toteuttaa käyttämällä modaaleja pop-up-ikkunoita (32), koska niiden käyttäjäystävällisyys, ulkoasu ja informaatioarvo olivat niiden vaihtoehtoisia tekniikoita parempia(kuva 5.) ASP.NETin komponenttikirjaston lisäksi päätin käyttää hyväkseni Piimegalla jo käytössä olevaa Telerikin UI-kirjastoa (27), koska sen tekniikat ja ulkoasu soveltuivat mielestäni paremmin edessäni olevaan haasteeseen. Telerikin käyttö mahdollisti samalla myös helpon Ajax-käsittelyn ja se on minulle tutumpi komponenttikirjastona sekä oman näkemykseni mukaan muodollisesti pätevämpi Microsoftin peruskomponentteihin verrattuna.



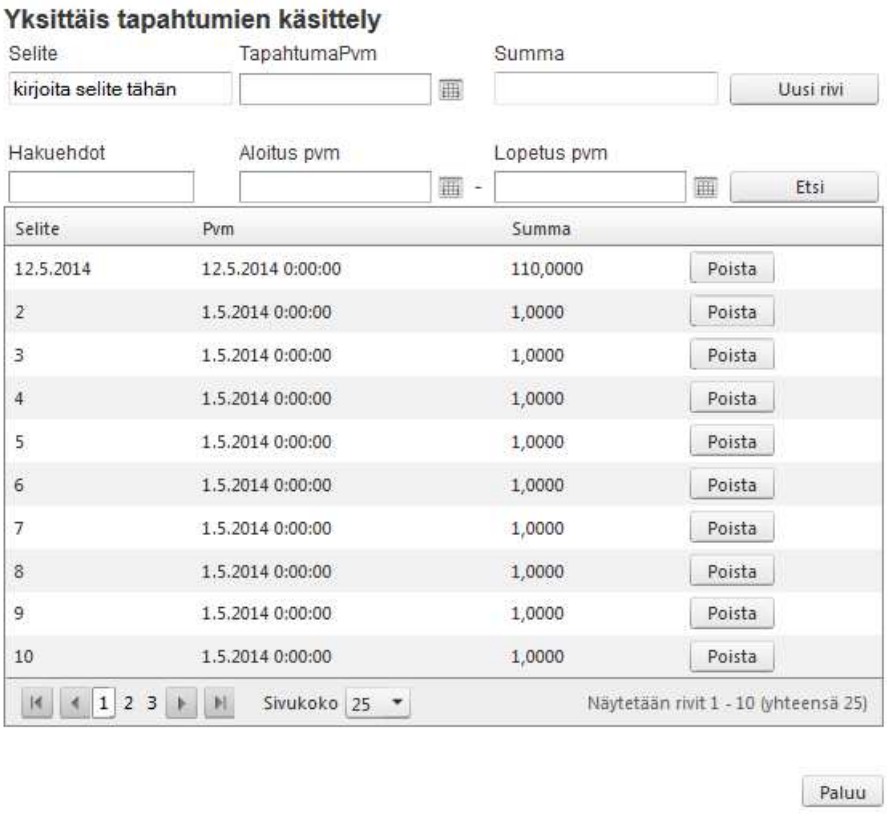
Selite	10/2014	11/2014	12/2014	13/2014	14/2014	15/2014	16/2014	17/2014	18/2014	19/2014	20/2014	21/2014	22/2014	23/2014	24/2014	25/2014	26/2014	27/2014	28/2014	yht:
Provisiot	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-5482,30	0	-5482,30
Palkkakulut	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,00
Asentajien - palkat	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,00
Projektikulut	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1240,00	0	0	0	1240,00
Myyntilaskut	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,00
Kerta tapahtuma	0	0	0	0	0	0	30000,00	0	23,00	0	110,00	0	0	0	0	0	0	0	0	30133,00
kaksi toistoa	0	0	0	0	0	0	955,00	955,00	0	0	0	0	0	0	0	0	0	0	0	1910,00
vk kokoajan juoksee	0	0	0	0	0	0	0	0	30,00	30,00	30,00	30,00	30,00	30,00	30,00	30,00	30,00	30,00	30,00	330,00
vk16 x3 x150	0	0	0	0	0	0	150,00	150,00	150,00	0	0	0	0	0	0	0	0	0	0	450,00
Testi vol 3	0	0	0	0	0	0	0	0	0	0	1000,00	1000,00	1000,00	0	0	0	0	0	0	3000,00
	155,50	155,50	155,50	155,50	155,50	155,50	31260,50	32365,50	32568,50	32598,50	33738,50	34768,50	35798,50	35828,50	35858,50	37128,50	37158,50	31706,20	31736,20	31736,20

KUVA 1. Ulkoasu, Etusivu.

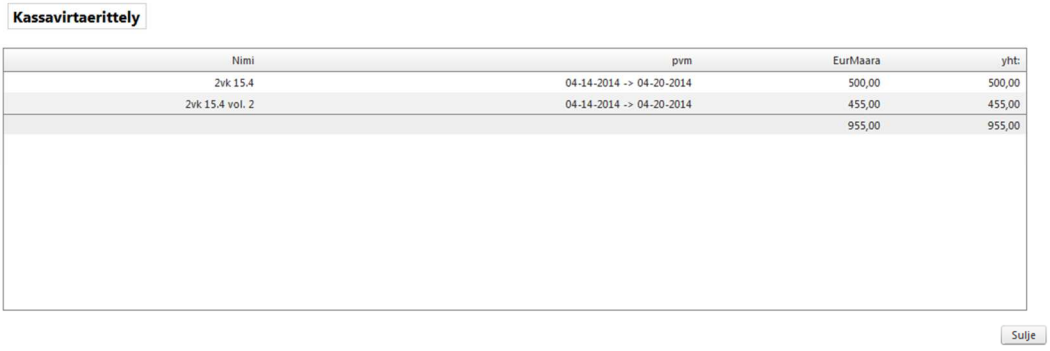
KUVA 2. Ulkoasu, Toistuvan tapahtuman käsitteleminen.



KUVA 3. Ulkoasu, Toistuvan rivin käsittely.



KUVA 4. Ulkoasu, Yksittäistapahtuman käsittely.



KUVA 5. Ulkoasu, Erittely.

3.2 Kassavirtojen muodostuminen

Kassavirrat muodostuvat yrityksen tuloista ja menoista, ja esittävät todellista dataa yrityksen taloudellisesta suunnasta. Tässä tapauksessa tulot ja menot ovat jo järjestelmässä. Tästä poikkeavassa tilanteessa ne syötetään

järjestelmään kiinteinä tai yksittäisinä kuluina. Kyseessä olevat tulot ja menot vaikuttavat laskelmaan ainoastaan jos ne sijoittuvat aikajanalla seurattavaan otantaan. Tästä poikkeuksena ovat kiinteät tapahtumat, mikäli niiden toistuvuus ylittää otannan raja-arvon, vaikka niiden luomishetki olisi tapahtunut seuratun ajan ulkopuolella.

3.3 Kassavirtaennuste

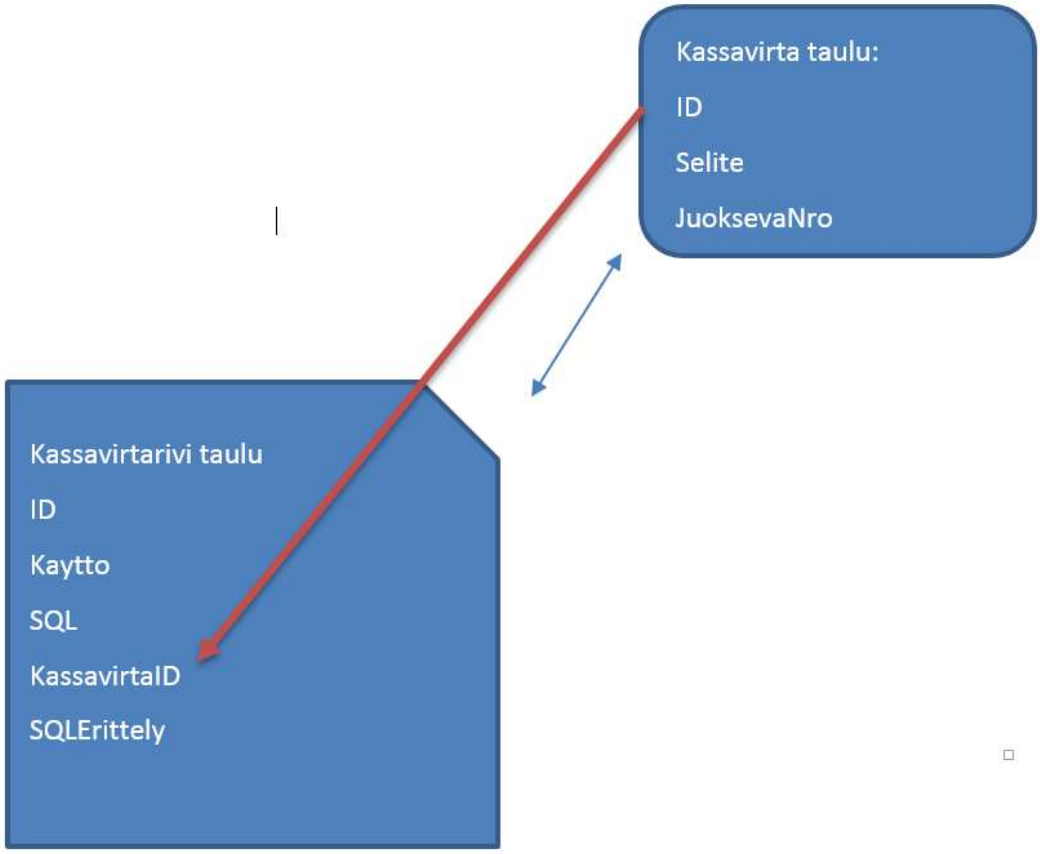
Itsessään kassavirtojen ennustaminen päätettiin toteuttaa yksinkertaistetusti niin sanottuna lyhyen ajan ennusteena, ilman että rakennettaisiin todellista ennustealgoritmia ennustamaan tulevien vuosien tapahtumia. Suunnitteluvaiheessa päätettiin keskittyä asiakkaille oleellisemmassa asemassa olevien kassavirtojen ennustamiseen. Edellä mainituissa tarkastellaan tietokannassa jo olemassa olevia rahavirtoja ja rakennetaan sen perusteella näkymä, joka esittää tuon kyseisen datan selkeässä muodossa annetulla aikavälillä. Näin saatiin tuotua esille mahdollisimman todenmukainen kuva yrityksen tulo- ja menovirroista.

4 TIETOKANTARATKAISUT

Piimega Total ohjelmistokomponentteineen tarvitsee toimiakseen tietokantapalvelimen ja toimivan palvelinyhteyden. Kassavirtaennustekomponentissa esitetyt tietoa esittävät kuvaajat rakennetaan tietokannan tietojen perusteella. Seuraavassa luvussa käsitellään tietokantaratkaisuja koskien tiedonhakua ja tiedonsyöttöä.

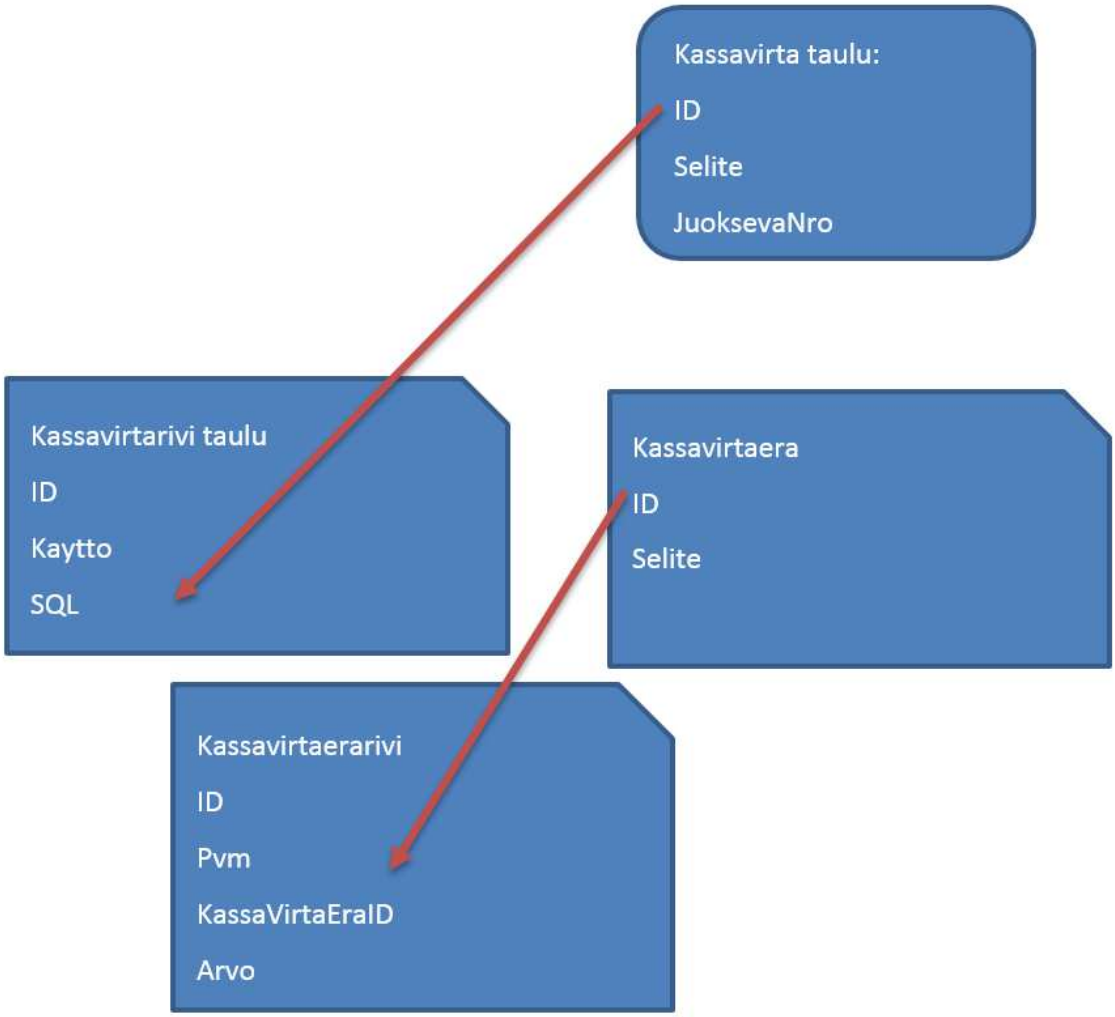
4.1 Taulut

Kassavirtaennusteen on tarkoitus yhdistää suuria määriä tietoa helposti luettavaan muotoon. Tämän seurauksena pääosa esitettävistä tiedoista joudutaan hakemaan eri tauluista. Kassavirtaennusteelle rakennettiin hierarkiataulurakenne, jossa päätaulu pitää sisällään primääriavaimet ja hakutyypit. Näiden avulla saadaan sidottua hierarkiassa alempana olevat taulut hierarkkiseen tiedonhakuun. (Kuva 6.)



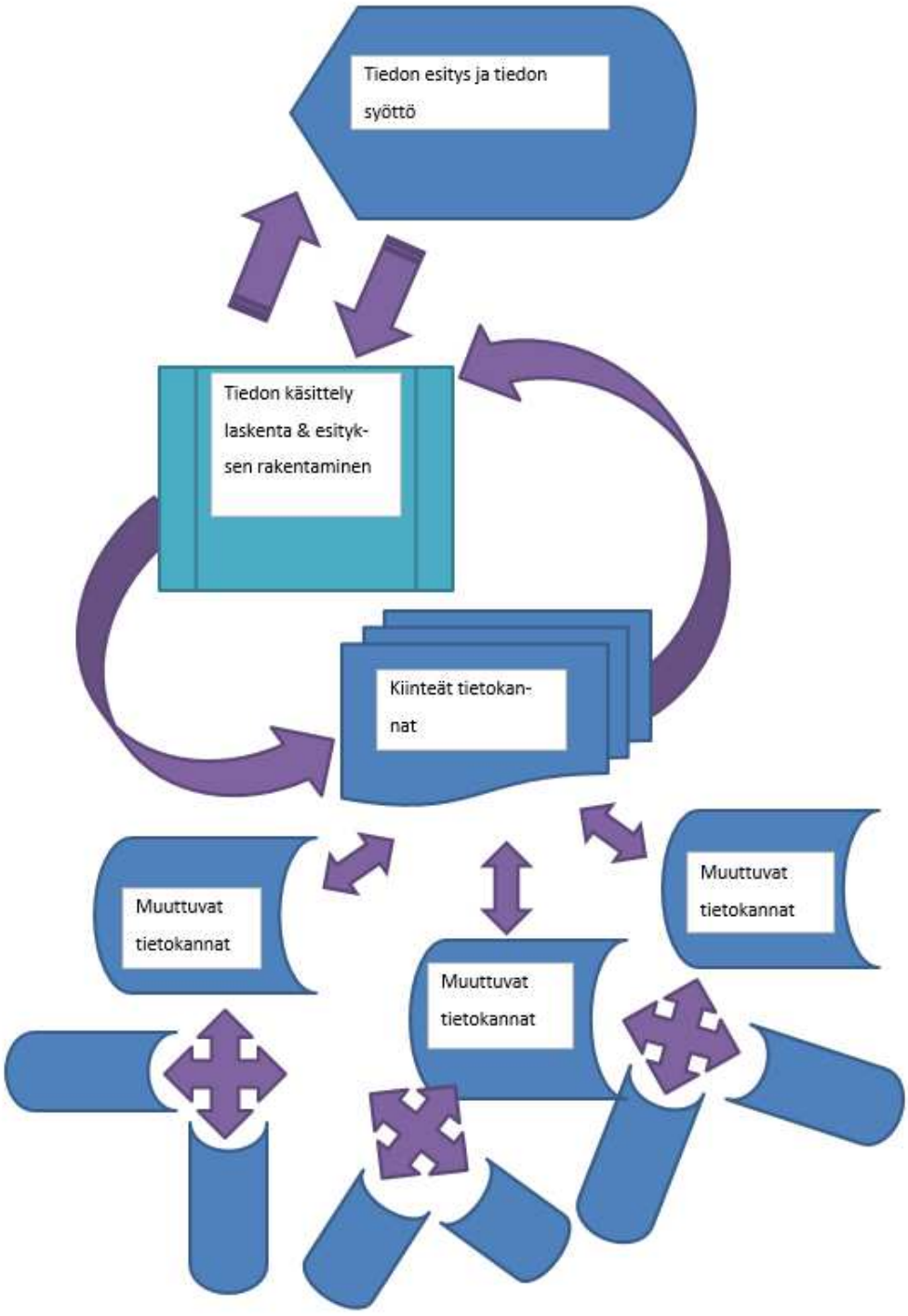
KUVA 6. Alkuperäinen taulurakenne

Myöhemmin kuitenkin huomattiin, että tarvitaan kaksi lisätaulua käsittelemään kiinteitä, käyttäjien syöttämiä tapahtumia (Kuva 7).



KUVA 7. Taulurakenne

Näiden neljän perustaulun avulla järjestelmä on yhteydessä muihin tauluihin ja rakentaa esityksen RadHtmlChartiin sekä RadGridiin (Kuva 8).



KUVA 8. Järjestelmän toiminta

4.2 Tiedonhakuratkaisut

Suunnitteluvaiheessa tultiin siihen tulokseen, että tiedonhaun tulisi toimia siten, että se ei olisi sijoitettuna ohjelmistokoodiin, vaan tietokantahakujen pitää olla sijoitettuna tietokantaan. Tähän ratkaisuun päädyttiin, koska asiakasyritysten tietokannat eroavat toisistaan ja kassavirtaennusteella on potentiaalia laajeta muihinkin yrityksen ohjelmistoihin. Toisin sanoen tiedon haku suoritetaan isätauluun, jossa sijaitsevien SQL-lausekkeiden avulla tehdään uusia hakuja tauluihin, joissa itse data sijaitsee. Tämä toimintatapa mahdollistaa ohjelmistokomponentin helpon uusiokäytön Piimegan sisällä sekä helpottaa asiakaskohtaista kustomointia.

4.3 Tiedonsyöttöratkaisut

Kassavirtaennusteen tietojen syötön ohjelmoinnissa törmäsin pieneen ongelmaan. Koska tietokantaratkaisut eriävät asiakasyritysten kesken, tiedon sijoitus syöttötilanteissa täytyi rakentaa staattisesti, eli toisin sanoen kassavirtaennusteelle tarvittiin pari uutta tietokantataulua. Näihin sijoitettiin niin sanottuja kiinteitä kuluja ja yksittäistapahtumia, jotka ovat kerran tapahtuvia kiinteitä kuluja. Tällä haluttiin varmistaa, että ohjelmistokomponentti olisi mahdollisimman luotettava ja informatiivinen sekä mahdollistaisi mahdollisimman kattavan palvelukokonaisuuden asiakasyritysten käyttöön. Tämä toteutettiin käytännössä rakentamalla kaksi taulua: isä- ja lapsitaulu, joista edellä mainittu pitää sisällään ID:n ja nimen ja johon lapsitaulut sitten sidotaan. Juuri tuon isätaulun primääriavaimen (20) (ID:n) avulla kyseiset tiedot sitten parseroidaan (19) näytölle sopivaan muotoon. Lapsitaulussa on arvo, arvon selite ja toistomäärä, jonka perusteella kyseinen kulutapahtuma rakennetaan normaalin SQL-haun avulla rakennetun taulun jatkeeksi.

5 TIETOJENKÄSITTELY JA ESITTÄMINEN

Päätin ohjelmoida Telerikin RadGridin tietojen esityksen lähdekoodin puolelta. Tällä pyrin saavuttamaan etua mark-upin (13) puolella rakennettuun taulukkoon ja sekarakennettuun taulukkoon verrattuna. Lähdekoodin puolella rakennettu taulukko mahdollistaa ajon aikana luodut sarakkeet ja tapauskohtaiset muutokset sekä poistot, siinä missä sekarakennettu taulukko estäisi sarakkeiden poistamisen ajon aikana. Tämän vuoksi dynaamisesti rakennetun taulukon toteuttaminen muuttuisi mahdottomaksi. (33; 34.)

Lähdin rakentamaan koodia käsittelemällä tietokantataulun prototyypin while-silmukalla, jolla käsitellään päiviä, viikkoja ja kuukausia aikaotannan puitteissa. Tämän prototyypin pohjalta aloin rakentamaan järjestelmää, joka rakentaa taulukkoon sarakkeita vastaamaan tietokannasta haettua oikeaa dataa. Teknisesti tämä toteutettiin for each -silmukoissa, joissa datan primääriavaimet ja kolumnien nimet asetettiin kohdilleen.

Tällä menetelmällä saavutettiin se etu, että sarakkeiden nimet ja datan primääri-avaimet oli mahdollisimman helppo sovittaa yhteen. Jokainen tällä tekniikalla luotu sarake tulee kuitenkin poistaa yksitellen, koska Telerikin dynaamisten kolumnien luonnissa esiintyvä ominaisuus ei mahdollista kaikkien sarakkeiden poistamista kerralla (34). Toisin sanoen, jos taulukossa on sarakkeita, ajetaan sarakkeiden luontivaiheessa sarakkeiden poisto while-silmukassa alaspäin iteroinnilla. Näin saavutetaan varmempi toiminnallisuus ja vältetään indeksin poiston aikainen risteäminen, joka tapahtuu kun poistettava indeksi ja ajettava indeksi eroavat kooltaan. Tämä aiheuttaisi ajonaikaisen virheen, joka taas aiheuttaisi kaatumisen. Koska risteäminen ei tapahdu joka tilanteessa, voisi virhe aiheuttaa turhaa ajanhukkaa ylläpitovaiheessa.

Kaikki kassavirtaennusteen kulut ja menot eivät löytyneet suoraan asiakasyritysten tietokannoista. Käyttökokemuksen parantamiseksi mahdollistettiin tietokannan luontivaiheessa tietojen syöttö asiakkaille.

Koska SQL:n ajon aikainen luominen olisi ollut riskialtista, varmatoimisuus kyseenalaista ja sen luominen aikaa vievää, se toteutettiin luomalla kaksi taulua, joihin kirjoitetaan asiakkaan kulloinkin haluamat arvot ja niiden toistuvuus.

Asiakas valitsee käyttöliittymän vetovalikosta joko yksittäis- tai toistuvan kulun. Mikäli asiakas valitsi toistuvan kulun, annetaan asiakkaalle mahdollisuus luoda haluamansa niminen rivi. Tässä yksinkertaisessa esimerkissä olkoon tuon rivin nimi Palkat. Kun Palkat-rivi on luotu, asiakkaalle annetaan mahdollisuus luoda lapsitaulun rivi, tässä tapauksessa Jorman Palkka. Edellä mainittu lapsirivi määrittää esitettävän toiston tyypin (määräaikainen tai toistaiseksi voimassa oleva), toiston arvon, toiston aloituspäivämäärän sekä toiston arvon toistuvuuden (kuukausittainen tai viikoittainen).

Asiakkaan lisättyä käyttöliittymästä rivit Palkat ja Jorman Palkka ajetaan postpack (21), joka johtaa tietojen uudelleenhakuun ja esityksen uudelleenrakennukseen ja piirtoon. Komponentin rakentaessa esitystä tarkistetaan onko kiinteitä tai yksittäistapahtumia olemassa, kuten tässä esimerkki tapauksessa. Perustietokantarivien haun jälkeen aletaan käsittelemään kyseiset rivit tauluun, joka toimii RadGridin tietolähteenä.

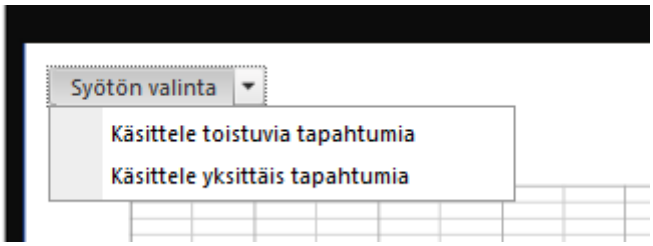
Tämä tapahtuu erillisen funktion sisällä, joka ottaa vastaan käsittelemättömän tietotaulun ja palauttaa käsitellyn tietotaulun. Funktion sisällä haetaan kassavirran kiinteät kulut ja parsitaan ne tietokannan tietojen jatkoksi.

Jokaisella kiinteällä kulurivillä on tiedoissaan sen toistotyyppi (kuukausi tai viikko), toistomäärä (tietty määrä toistoja tai toistaiseksi voimassaoleva), aloituspäivä (milloin toisto aloitetaan) ja summa, jota toistetaan. Kiinteiden kulujen tietojen esittäminen rakentuu seuraavasti. Ensimmäiseksi katsotaan otantatyypin (viikko tai kuukausi) ja tarkistetaan sen pituus, sen jälkeen käydään while-silmukassa lävitse jokainen kiinteä kulu. While-silmukan sisällä ensimmäiseksi tarkistetaan, mitä tyyppiä kiinteä tapahtuma on, ja tämän johdosta päädytään for each -silmukkaan. Silmukka on rakennettu käsittelemään kyseisen tyypin tapahtuma kuukausi tai viikko-otannan jatkoksi. For each -silmukan sisällä verrataan kiinteän tapahtuman aloittamishetkeä

otannan aloitushetkeen ja päätellään if-rakenteella, onko kyseinen kiinteä tapahtuma jo alkanut pyöriä ennen otantaa. Tämän tiedon perusteella kasvatetaan sijoittaja-integeriä, jonka avulla mahdollisesti sijoitetaan tapahtuma tietotauluun. Silmukan pyöriessä tarkistetaan, ettei toistojen määrä ylity. Luonnollisesti kuukausittaisesitystavassa joudutaan viikottaistapahtumat kertomaan 1–5 kertaa riippuen ISO8601-standardin (9) mukaisesta kalenteriviikkojen määrästä. Edellä mainitun periaatteen mukaan kuukausittaiset kulut toistetaan viikkonäkymässä 1–5 viikon välein, jotta standardin mukaisuus säilyy.

Tarkempi tietojenkäsittely ja -esittäminen toteutettiin käyttämällä ASP.NETin paneeleja, jotka tuotiin modaalisesti esille ajaxin avulla. Tällä tekniikalla toteutettiin niin tietojen syöttö kuin tietojen esitys. Modaalisen tietojen esityksen etuna on informatiivisuus, sillä se piilottaa käyttäjältä turhan tiedot, jolloin käyttäjä voi keskittyä oleellisimpaan dataan.

Tietojen syöttö toteutettiin ASP.NET -paneeleilla (28), joihin pääsee käsiksi list-buttonilla (Kuva 8).



KUVA 8. Ulkoasu, syötön listbutton.

Buttonilla voidaan määrittää käsitelläänkö yksittäistapahtumaa vai toistuvaa tapahtumaa, jolloin tuodaan esille RadGrid, josta voidaan valita ja muokata jo olemassa olevia rivejä. Kun käyttäjän valittaessa on jo olemassa oleva rivi, avautuu uusi modaalinen ikkuna, josta löytyvät lapsirivien syöttö ja lapsirivien tiedon esitys uudessa RadGridissä.

Kassavirran tapahtumien tietojen esittäminen toteutettiin sisällyttämällä tietokantatauluihin NCHAR (18) eli kenttä, jossa sijaitsee tietokantahaku.

NCHARilssa sijaitsevalla hakulauseella haetaan tapahtuman tiedot annettujen aikaparametrien mukaan, jolloin kyseisen solun valitseminen avaa modaalisen ikkunan, jossa esitetään tietorivit, joista solussa oleva kokonaissumma koostuu.

Valitettavasti kiinteiden tapahtumien kohdalla näin yksinkertainen ja tehokas ratkaisu ei ollut mahdollinen, vaan jouduttiin rakentamaan hakujärjestelmä, joka hakee kiinteistä kuluista solulle sijoitetut tiedot niiden isä-ID:n ja ajankohdan perusteella. Pitkän taistelun jälkeen menetelmä jouduttiin rakentamaan käyttämällä tiedonvälitystä JavaScriptillä (12), jonka avulla lähetetään client sidestä server sidelle (6; 25) solun XY-koordinaatit custom postpackissa. XY -koordinaattien avulla saatiin paikallistettua oikea ID ja parsittua kasaan oikea ajankohta. Näiden tietojen perusteella suoritetaan SQL-haku, joka pois sulkee turhat elementit jättäen jäljelle vain kyseiseen soluun sijoitetut elementit. Poissuljenta toteutettiin käyttämällä vastaavaa ratkaisua kuin tiedonsyötössä, jossa määriteltiin arvon oikea sijoituspaikka alkupäivän ja toistojen perusteella. Nyt sama tehtiin tietokannallisesti.

6 OPINNÄYTETYÖ PROSESSINA

Projektin suunnittelu aloitettiin pitämällä suunnittelupalaveri, jossa merkittiin ylös asioita, joita asiakas tuotteeseen halusi. Sen lisäksi listattiin ylös ominaisuuksia, joita tuotteessa tulisi olla, jotta se olisi myös käytettävissä muissa yrityksen järjestelmissä. Näiden seikkojen lisäksi huomioon tuli ottaa myös muut mahdolliset asiakkaat. Kun järjestelmän rautalankamalli alkoi olla kassassa, otin yhteyttä koululle ja sovin palaverin työn tilaajan ja ohjaavan opettajan kanssa.

Projektin työn aikainen dokumentaatio pidettiin minimissä, ja pääosin toimin Excel-dokumentin pohjalta, jossa minulla oli ylhäällä halutut toiminnollisuudet ja niiden sisältämät ominaisuudet sekä lyhyt kuvaus niiden halutusta toiminnasta. Tämän lisäksi piirsin käyttöliittymän rautalankamallin itselleni, jotta pystyisin työskentelemään mahdollisimman nopeasti toteuttaessani visiotani.

Projektia toteuttaessani tulin siihen tulokseen, että asiakkaan tarpeiden laajempi kartoitus olisi ollut suotavaa, sillä uusia ominaisuuksia koskevien vaatimusten tipahteleminen asiakkaalta oli jokseenkin opettavaista. Jälkiviisaana on helppo todeta, että esimerkiksi paperimallilla toteutettu käyttäjätestaus ja ominaisuuksien kirjaaminen ylös olisi säästänyt työaikaa. Kassavirtaennusteen pääominaisuuksien ollessa valmiita tuote esiteltiin asiakkaalle kolmannen tahon toimesta, eli en itse ohjelmoijana ollut osallisena kummassakaan määrittelykierroksessa. Tämä toimintamalli oli mielestäni virheellinen. Mielestäni ohjelmoijan olisi pitänyt olla paikalla esittelyssä, jolloin uusien ominaisuuksien määrittely olisi ollut selkeämpää. Kaiken kaikkiaan kassavirtaennuste kuitenkin ohjelmoitiin suunnitellusti, vaikkakin kaikkia määrittelyitä ei saatu aikataulutuksen puitteissa toteutettua.

Työn toteutuksen seurauksena yrityksen asiakkaalla on käytössään kassavirtaennuste, joka esittää asiakkaan kassan graafisesti ja taulukkona (Kuvat 1–5). Sen lisäksi asiakkaalla on mahdollisuus syöttää kiinteitä kassatapahtumia ja yksittäiskassatapahtumia graafisen käyttöliittymän kautta. Asiakas saa myös lisätietoa tapahtumista valitsemalla haluamansa taulukon solun aktiiviseksi. Tämä selkeyttää sitä, mistä tietty kassatapahtuma koostuu.

Työn tuloksena valmistui myös luonnollisesti valtava määrä SQL-haku- ja luontikoodia, joka oli oleellinen osa järjestelmän toimintaa.

8 POHDINTA

Opinnäytetyön tarkoitus oli suunnitella ja luoda uusi ohjelmistomoduuli Piimega Oy:n Total ohjelmistoon. Moduulin nimi oli Kassavirtaennuste. Kassavirtaennustemoduulilla tarkoitetaan tässä tapauksessa rahavirtojen seurantaa annetulla ajanjaksolla. Ohjelmistomoduuli toteutettiin asiakkaan vaatimusten mukaisesti, mutta siihen jäi vielä kehitettävää. Jälkiviisaana on helppo huomata kuinka, ohjelmisto olisi kuitenkin kannattanut ohjelmoida. Seuraavassa kappaleessa käsittelen asioita, jotka mielestäni olisi kannattanut tehdä toisin.

Tietokannan kiinteiden tapahtumien taulurakenne olisi ollut järkevämpi toteuttaa yksinkertaistetummin, hyödyntäen useampaa taulua. Nykyisellään järjestelmä toimii siten, että isärivien perusteella voidaan päätellä lapsirivien määrä ja toteuttaa ne esitykseen virtuaalisesti. Omasta mielestäni järjestely on uudelle ylläpitäjälle hankalasti ymmärrettävä. Sen lisäksi toimintatavalla ei ole huomattavaa etua kahden erillisen taulun tietokantarakenteeseen.

Taulukkorakenteen muutos olisi myös voinut yksinkertaistaa kiinteiden tapahtumien parsimista, sillä mikäli olisi käytetty kahta tietokanta taulua ja luotu uudet tapahtumat omiksi tietokantariveikseen, niiden parsiminen tiedon esitykseen olisi ollut huomattavasti helpompaa. Ylläpidollisesti järjestely olisi huomattavasti kevyempi kuin nykyinen järjestelmä. Ohjelmistokomponentti toimii kuitenkin asiakkaan asettamien vaatimusten mukaisesti, ja se täyttää kassavirtaennusteen tunnusmerkit.

Tulevaisuudessa kassavirtaennuste voitaisiin laajentaa käsittelemään pitemmän aikavälin tapahtumia menneiltä vuosilta. Näiden perusteella voitaisiin luoda pitempiaikaista ennustetta, joka ei olisi pelkästään sidoksissa syötteisiin, vaan antaisi lisäksi tietoa siitä, mitä tähän aikaan on tapahtunut menneinä vuosina. Tällä voitaisiin ennustaa mm. asiakkaiden ostokäyttäytymistä eri vuodenaikoihin.

Tämän lisäksi olisi täysin mahdollista laajentaa kassavirtaennustetta sellaiseen suuntaan, että sen nykyisten tai tulevaisuudessa laajennettujen tietojen lisäksi siinä esitettäisiin menojen budjetti. Tällä lisäyksellä saataisiin aikaan

kokonaisvaltaisempi katsaus taloustilanteeseen, joka taas helpottaisi asiakasyrityksen johtoportaan toimintaa.

Monesti sanotaan, että opinnäytetyön tekemisen tulisi olla pelkkä esitys osaamisesta, eikä oppimiskokemus. Itse olen tässä asiassa eri mieltä, sillä pääsin kassavirtaennustetta rakentaessani oppimaan uusia asioita ja kertaamaan jo osaamiani asioita.

Teknisesti työ oli mielestäni haastava, sillä sen rakentamiseen tarvittiin laajaa ymmärrystä tietokannoista sekä ASP.NET web-tekniikasta. Työn haastavimmaksi osuudeksi näkisin kiinteiden tapahtumien parsimisen, sillä kyseisellä tavalla rakennettuna kassavirtaennusteen kiinteät tapahtumat olivat todella vaikeita saada toimimaan 100 %:n toimintavarmuudella.

Työtä tehdessäni huomasin, että selkeä ja muuttumaton suunnitelma olisi helpottanut työskentelyä huomattavasti. Selkeän suunnitelman puuttumisesta syytän omaa kokemattomuuttani, sillä minun olisi pitänyt vaatia selkeämmin pääsyä osalliseksi työn ominaisuuksien kartoittamiseen sekä sen esittelyyn asiakkaalle. Lisäksi jälkiviisaasti minun olisi pitänyt tuoda esille mahdollisuus käyttää esimerkiksi yksinkertaistettua paperimallia asiakaskartoituksessa, ennen itsejärjestelmän ohjelmoinnin aloittamista. Uskoisin, että näillä muutoksilla olisi säästetty huomattavasti kehitysaikaa.

LÄHTEET

1. Getting Started with the .NET Framework. 2014. Microsoft. Saatavissa: http://msdn.microsoft.com/en-us/library/hh425099.aspx. Hakupäivä 18.11.2014.

2. .NET Framework. 2014. Wikipedia. Saatavissa: http://fi.wikipedia.org/wiki/.NET_Framework. Hakupäivä 18.11.2014.

3. Gallo, Alessandro –Barkol, David –Vavilala, Rama 2008. ASP NET AJAX in Action. Manning Publications. Viitattu 18.11.2014.

4. Garret, James 2005. Ajax: a New Approach to Web Applications. Saatavissa: http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications/. Hakupäivä 28.11.2014.

5. Esposito, Dino 2008. Programming Microsoft ASP.NET 3.5.

6. Client-Side. 2014. Wikipedia. Saatavissa: http://en.wikipedia.org/wiki/Client-side. Hakupäivä 20.11.2014.

7. Piimega Total Erp. 2014. Piimega. Saatavissa: http://www.piimega.fi/tuotteet/toiminnanohjausjarjestelma/piimega-total-erp? SID=U. Hakupäivä 19.11.2014.

8. For Each...Next Statement (Visual Basic). 2014. Microsoft. Saatavissa: http://msdn.microsoft.com/en-us/library/5ebk1751.aspx. Hakupäivä 20.11.2014.

9. ISO Date and time format – ISO 8601. ISO. 2012. Saatavissa: http://www.iso.org/iso/home/standards/iso8601.htm. Hakupäivä 20.11.2014.

10. Int32 Structure .Microsoft. 2014. Saatavissa: http://msdn.microsoft.com/en-us/library/06bkb8w2.aspx. Hakupäivä 20.11.2014.

11. Integer Data Type (Visual Basic). 2014. Microsoft. Saatavissa: http://msdn.microsoft.com/en-us/library/system.int32.aspx?cs-save-lang=1&cs-lang=vb#code-snippet-1. Hakupäivä 20.11.2014.

12. Javascript. 2014. Mozilla Corporation. Saatavissa: https://developer.mozilla.org/en-US/docs/Web/JavaScript. Hakupäivä 18.11.2014.

13. Markup language. 2014. Webopedia Vangie Beal. Saatavissa: http://www.webopedia.com/TERM/M/markup_language.html. Hakupäivä 28.11.2014.

14. SQL Server Tutorial – Learn Microsoft SQL 2012. w3computing. Saatavissa: http://www.w3computing.com/sqlserver/. Hakupäivä 18.11.2014.

15. Introducing Visual Studio. Microsoft. 2014. Saatavissa: http://msdn.microsoft.com/en-us/library/fx6bk1f4%28v=vs.90%29.aspx. Hakupäivä 18.11.2014

16. Modal Window. Wikipedia. 2014. Saatavissa: http://en.wikipedia.org/wiki/Modal_window. Hakupäivä 27.11.2014.

17. Window – Modal Pop Up. 2014. Telerik. Saatavissa: http://demos.telerik.com/aspnet-ajax/window/examples/modalpopup/defaultcs.aspx. Hakupäivä 19.11.2014.

18. What's the difference between CHAR and NCHAR data types and when do I use them. 2014. Sql Server Helper. Saatavissa: http://www.sql-server-helper.com/faq/data-types-p01.aspx. Hakupäivä 20.11.2014.

19. Definition parse Margret Rouse. 2005. WhatIs. Saatavissa: http://whatis.techtarget.com/definition/parse. Hakupäivä 18.11.2014.

20. SQL PRIMARY KEY CONSTAINT. 2014. w3schools.com. Saatavissa: http://www.w3schools.com/sql/sql_primarykey.asp. Hakupäivä 19.11.2014

21. How postback works in asp net. 2011. Evagoras. Saatavissa: http://www.evagoras.com/2011/02/10/how-postback-works-in-asp-net/. Hakupäivä 20.11.2014.

22. ASP.NET Button Overview. 2014. Telerik. Saatavissa: http://www.telerik.com/help/aspnet-ajax/button_overview.html. Hakupäivä 25.11.2014

23. UI for ASP.NET AJAX Documentation Overview. 2014. Telerik. Saatavissa: http://www.telerik.com/help/aspnet-ajax/htmlchart-overview.html. Hakupäivä 28.11.2014.

24. About RadGrid for ASP.NET AJAX. 2014. Telerik. Saatavissa: http://demos.telerik.com/aspnet-ajax/grid/examples/overview/defaultcs.aspx. Hakupäivä 20.11.2014

25. Server-Side. Wikipedia. 2014. Saatavissa: http://en.wikipedia.org/wiki/Server-side. Hakupäivä 20.11.2014.

26. Laiho, Martti 1992 SQL Vapk-Kustannus.

27. UI for ASP.NET AJAX. Telerik. 2014. Saatavissa: http://www.telerik.com/products/aspnet-ajax.aspx#complete-set-of-features-and-controls. Hakupäivä 18.11.2014.

28. Panel Class introduction. Microsoft. 2014. Saatavissa: http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.panel%28v=vs.110%29.aspx. Hakupäivä 25.11.2014.

29. Visual Basic Language Specification 11.0. 2012. Microsoft. Saatavissa: http://www.microsoft.com/en-

us/download/details.aspx?displaylang=en&id=15039. Hakupäivä 18.11.2014.

30. Grundgeiger, Dave 2002 Programming Visual Basic NET.

31. While..End While Statemnt(Visual Basic). 2014. Microsoft. Saatavissa: http://msdn.microsoft.com/en-us/library/zh1f56zs.aspx. Hakupäivä 20.11.2014.

32. Window- Modal Popup. Telerik. 2014. Saatavissa: http://demos.telerik.com/aspnet-ajax/window/examples/modalpopup/defaultcs.aspx. Hakupäivä 28.11.2014.

33. UI for ASP.NET AJAX Creating a RadGrid Programmatically. 2014. Telerik. Saatavissa: http://www.telerik.com/help/aspnet-ajax/grid-programmatic-creation.html. Hakupäivä 21.11.2014.

34. radgrid removing column on runtime – Chandra Mohan. 2014. Telerik. Saatavissa: http://www.telerik.com/forums/radgrid-removing-column-on-runtime. Hakupäivä 21.11.2014.